XWT

Declarative UI for Eclipse

Yves YANG (Soyatec)

Soyatec
Open Solution Company

Contents

- What is XWT?
- Architecture
- XWT Fundamentals
- JFace integration
- Component and Data View Management
- Integration with Existing Application
- Binding and Data Binding
- Advanced features
- Conclusion
- Q&A

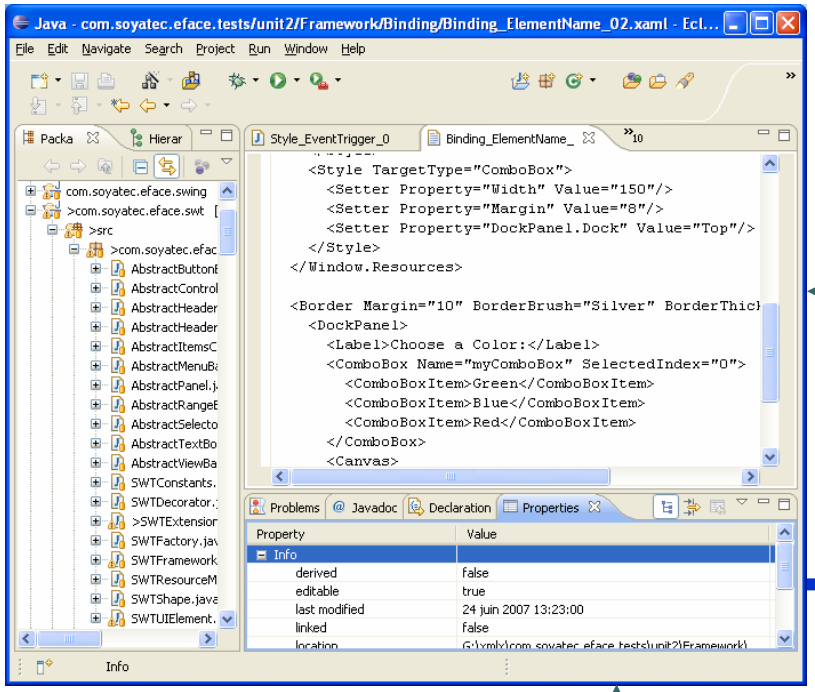
What is XWT?

XWT: **X**ML UI for S**WT**

- XWT is a XML dialect
- Designed for human-editable and tooling
- Dynamic mapping with programming model
- Complete declarative UI framework integrated with SWT/JFace, JFace data binding, etc.

Architecture

Concept



XML Files

GUI Ressource

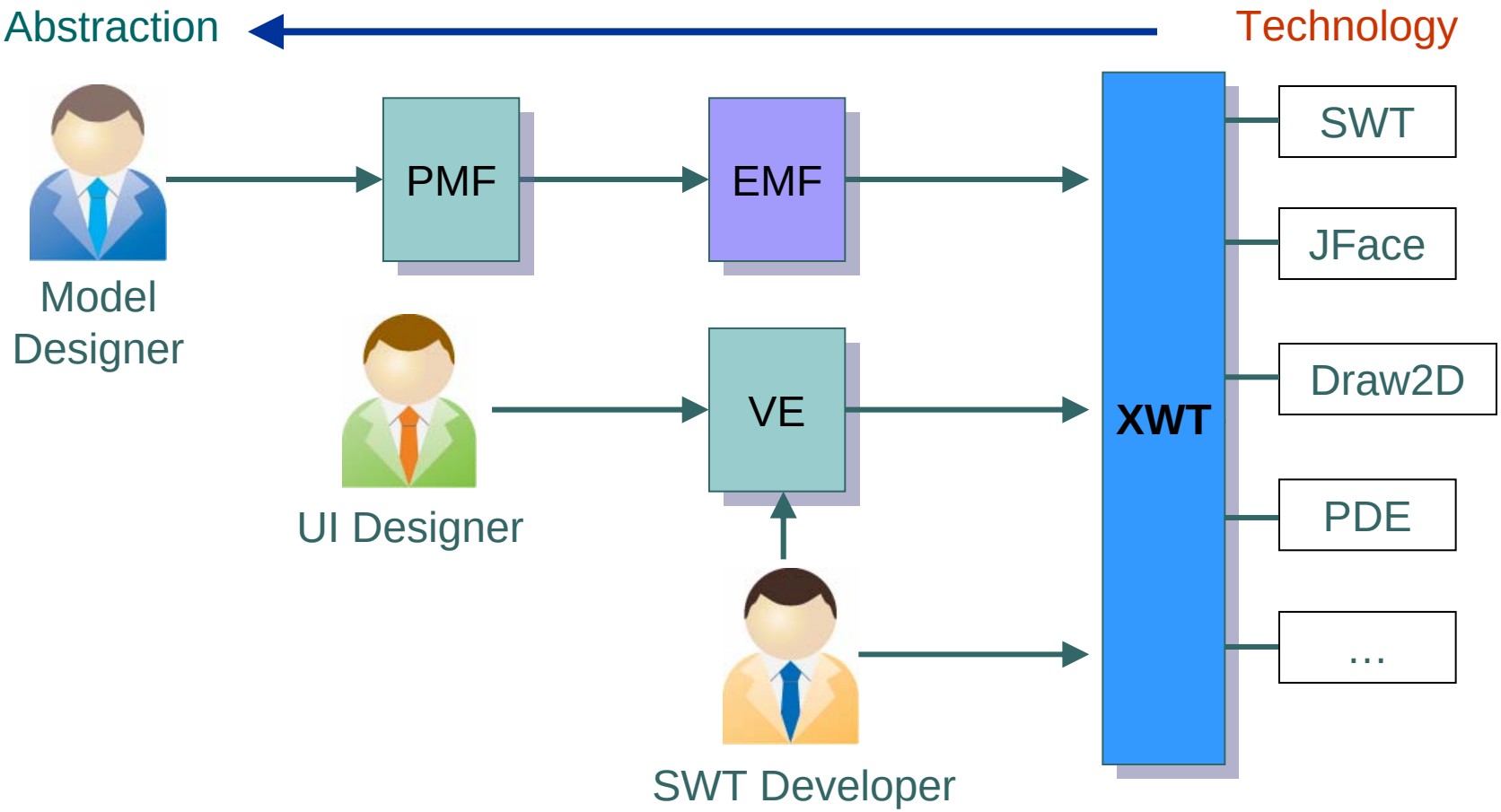
Event / Action

Java

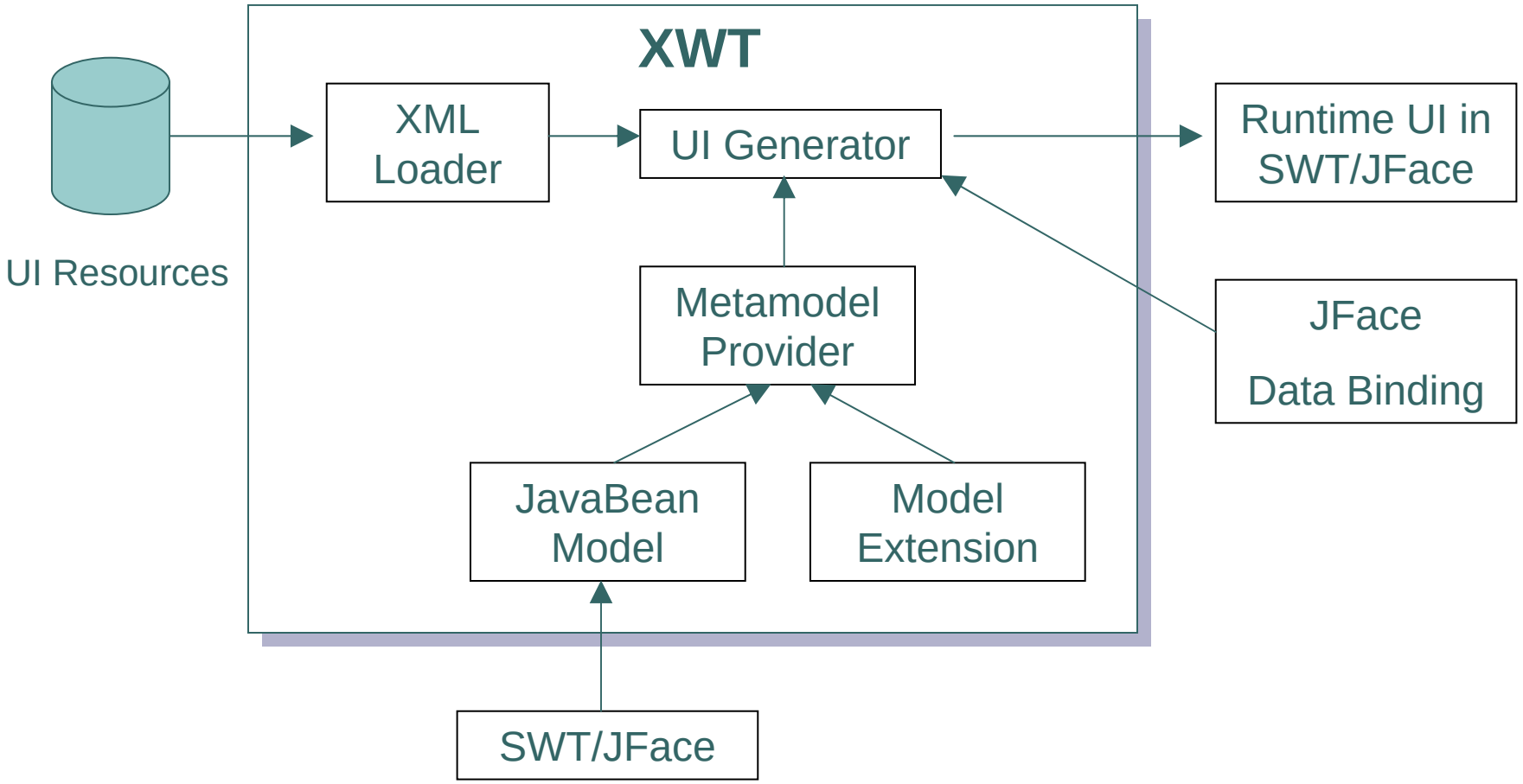
Dynamic GUI

Architecture

Position



Architecture



Architecture

- SWT Model as XWT model

  This model comes from the JavaBean reflection and additional models to enhance the limitations of SWT API.
- This model can be changed to another
- XWT provides more concepts on top of XML:
  - Resource separation between UI and Event handling
  - Dynamic mapping with UI Model and programming language
  - Resource management
  - Markup extensions
  - Binding expression language

XWT Fundamentals

Hello, world!

XML

```
<Shell xmlns="http://www.eclipse.org/xwt/presentation"
    Text=“Appearance">
   <Shell.layout>
      <FillLayout/>
   </Shell.layout>
   <Button Text="Hello world!"/>
</Shell>
```

Java

```
Shell shell = new Shell();
shell.setText("Appearance");
Shell.setLayout(new FillLayout());
Button button = new Button(shell, SWT.PUSH);
Button.setText("Hello world!");
```

XWT Fundamentals

Event Handling

XML

```
<Shell xmlns="http://www.eclipse.org/xwt/presentation"
       xmlns:x="http://www.eclipse.org/xwt"
   x:Class="ui.Handler"
   Text="Event handling">
   <Shell.layout>
      <FillLayout/>
   </Shell.layout>
   <Button SelectionEvent="onClick" Text="Click here"</>
</Shell>
```

Java

```
package ui;
import org.eclipse.swt.Event;
public class Handler {
   public void onClick(Event event) {
        System.out.println("Hello world!");
   }
}
```

XWT Fundamentals

Extensbility

- Any existing SWT components can be used directly
  - Declare a namespace for the package

    `xmlns:y="cls-namespace:ui"`
  - Prefix the class name by the namespace

    `<y:MyGridLayout/>`

```
<Composite
   xmlns="http://www.eclipse.org/xwt/presentation"
   xmlns:x="http://www.eclipse.org/xwt"
   xmlns:y="cls-namespace:ui">
   <Composite.layout>
        <y:MyGridLayout/>
   </Composite.layout>
   <Label text="Hello, world"/>
</Composite>
```

JFace integration

- JFace model is used directly
- LabelProvider/ContentProvider are handled directly by XWT class mapping
- Input data corresponds to Data Context

JFace integration


```
<Shell xmlns="http://www.eclipse.org/xwt/presentation"
       xmlns:x="http://www.eclipse.org/xwt"
       xmlns:j="clr-namespace:ui"
   Text="JFace integration">
   <Shell.layout>
      <FillLayout/>
   </Shell.layout>
   <ListViewer input="{Binding}”>
       <ListViewer.labelProvider>
           <j:PersonLabelProvider>
       </ListViewer.labelProvider>
       <ListViewer.contentProvider>
           <j:PersonContentProvider>
       </ListViewer.contentProvider>
   </ListViewer>
</Shell>
```

Binding & Data Binding

- XWT Binding Expression language
  - Data binding
  - Element binding

    Reference between element by name
- Based on Eclipse Data Binding engine
  - Data conversion
  - Data validation
  - Automatic update in two directions
    - Data source object to GUI
    - GUI modification to data source object
- Data context management

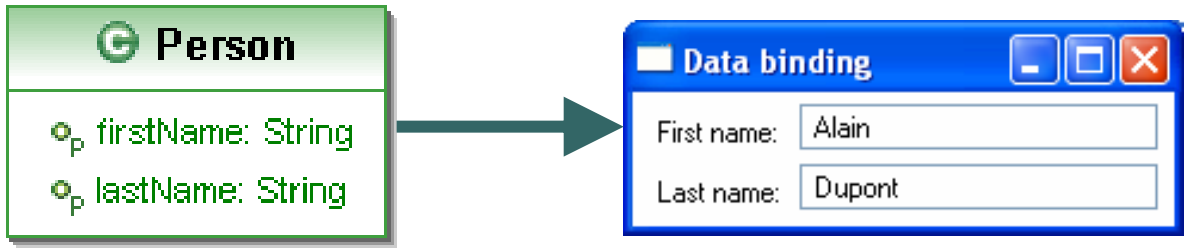
  Designed for View approach. Each View can be embedded into another.

Data Binding

XML

```
<Shell
   xmlns="http://www.eclipse.org/xwt/presentation"
   Title="Data binding">
   <Shell.layout>
       <GridLayout numColumns="2"/>
   </Shell.layout>

   <Label Text="First name:"/>
   <Text Text="{Binding Path=FirstName}"/>
   <Label Text="Last name:"/>
   <Text Text="{Binding Path=LastName}"/>
</Shell>
```



Component & Data View Management

- UI Component

  Define your UI components in XWT, instead of low level SWT in Java.

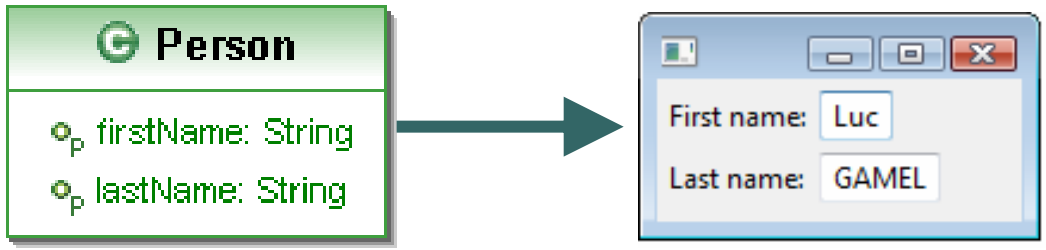
  - A View consists of a Java class for Event handling and a XWT resource for UI structure definition respectively.
  - Each component can be used directly as SWT Widget.
- Data View component
  - Each view is in fact a component that has an implicit data context

Component & Data View Management

XML

```
<j:PersonView
   xmlns="http://www.eclipse.org/xwt/presentation"
   xmlns:j=“clr-namespace:ui">
   <j:PersonView.layout>
       <GridLayout numColumns="2"/>
   </j:PersonView.layout>

   <Label Text="First name:"/>
   <Text Text="{Binding Path=FirstName}"/>
   <Label Text="Last name:"/>
   <Text Text="{Binding Path=LastName}"/>
</j:PersonView>
```

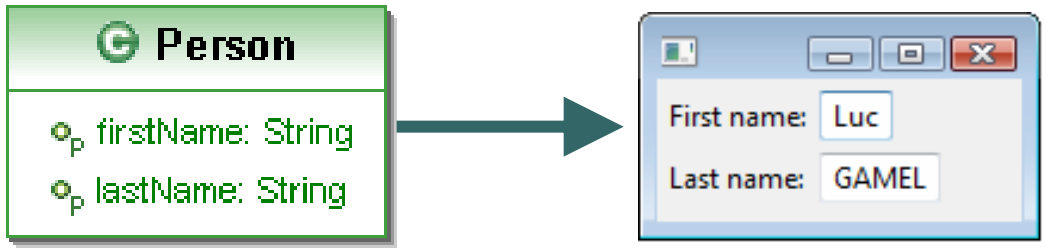


Component & Data View Management

```
package ui;
import org.eclipse.xwt.Composite;

public class PersonView extends Composite {
    public PersonView(Composite parent, int styles) {
       super(parent, styles);
    }

    // business logic and event handling
    …
}
```



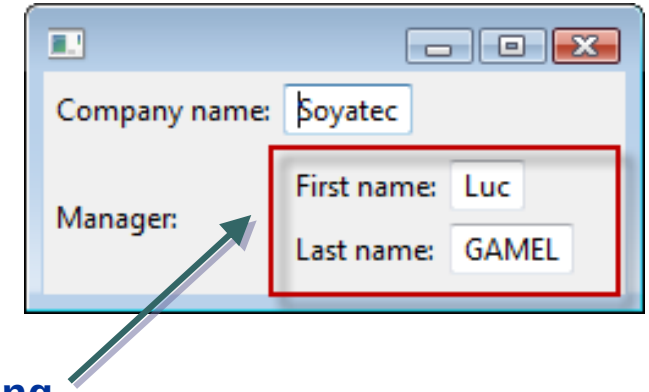
Component & Data View Management

XML

```
<j:CompanyView
   xmlns="http://www.eclipse.org/xwt/presentation"
   xmlns:x="http://www.eclipse.org/xwt"
   xmlns:j=“clr-namespace:ui“
   Text="Data binding">
   <j:CompanyView.layout>
       <GridLayout numColumns="2"/>
   </j:CompanyView.layout>

   <Label Text=“Company name:"/>
   <Text Text="{Binding Path=Name}"/>

   <Label Text=“Manager:"/>
   <j:PersonView x:DataContext="{Binding
        Path=Manager}"/>
</j:CompanyView>
```



Integration with Existing Application

The integration with existing application is really straightforward

- Import org.soyatec.xwt plugin
- Use the class XWT to load a resource in XML under a Composite

```
Control XWT.load(String file);
Control XWT.load(Composite container, String file);
Control XWT.load(Composite container, String file,
                 Object dataContext);
```

Further Features to implement

- Style

  CSS, Named Style and Style override
- Template for dynamic GUI

  Control template and Data template
- Script language support
- i18n support

Conclusion

- Very Lightweight solution (~110 Kbytes)
- High extensibility with existing Widgets
- Easy to integration with SWT application
- Straightforward solution for Eclipse developer

  UI programming language keeps SWT
- Complete features
  - Integration with WTP XML editor
  - Possible to integrate with VE

Issue

- SWT Model in XWT

  Using an abstract model is totally possible in XWT level. It is not a technical issue.

Q&A

Contacts

Yves YANG

CEO

Tel: +33 (1) 60 13 06 67

Email: yves.yang@soyatec.com

http://www.soyatec.com